

Minh Thanh Vo - 1251040  
Nguyen Ba Tuan Vu - 1251057

Chat Application

**CS494**

**INTERNETWORKING PROTOCOL**

Lecturer

Nguyen Tuan Nam

# overview

Application name: Chat Application.

Platform: Java application

Description:

Chat application is a simple chat app running on the TCP connection.

# FEATURES:

* User fill out the IP and Port to connect the server
* User can sign-up by username and password.
* User login by username and password
* User can see the online list user.
* User can send message to anyone in the list and all of user.
* Password is stored in file and encrypted with MD5 hash

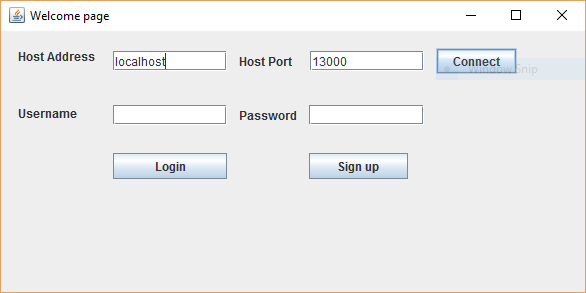
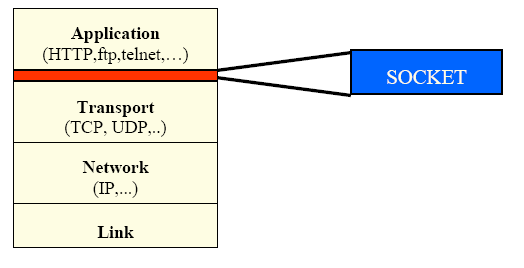


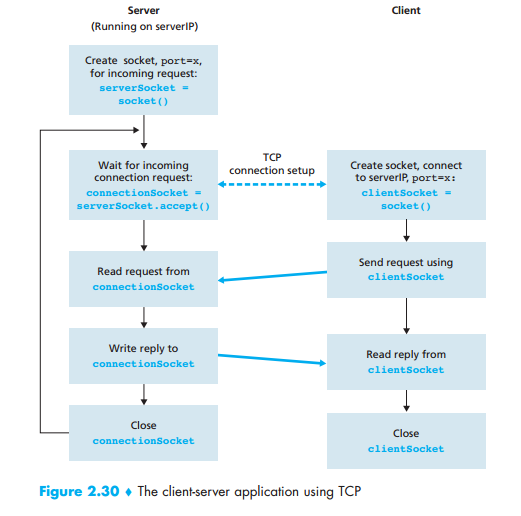
Figure 1: GUI Client

# STRUCTURE AND ALGORITHMS

## STRUCTURE



Chat Application is a socket application running on TCP connection. There are two main part: server and client talking to each other by messages sent through TCP connection.



* Server: has 2 sockets.
  + The first one is created when running the server. The role of socket is wait and listen all the connection coming from the client.
  + The second socket is created for taking the connection from the client. The role of socket is transferring and receiving the data between server and client.
  + After sending data, the connection is closed. Server continues to wait for the next other request.
* Client: create a socket, connect to server through by serverIP and port number. The client socket sent request to server and wait for accepting. After being accepted, the connection is made between server and client, so that client can send data to each other.

## algorithms

### server

The socket is organized with 5 classes:

* Database: handling the information of user, sign up new user and add information to file, checking user’s account.
* Message: data segment is wrapped and sent through TCP to client.
* ServerThread: each of server thread instanse represent a connection between client and server. Server thread listen for any message and hand it over to SocketServer to process. Also it forward messages from other users to the connected user. .
* SocketServer: containing main methods, including start ServerSocket for listening all the connection from clients and a list of ServerThread storing the information of client’s connection. The role of SocketServer is listening the request from the client, create a connection instanse and handling all the message from client.
* Server: main program, start a console program allow user to select port number, select the data account file and start server program.

The flow of running:

On the startup, the SocketServer runs in a separate thread. The role of it is waiting for connection and for each connection start a new thread ServerThread. Once the connection is established, ServerThread will listen for any message and hand it over to SocketServer to process. Its job will notify for user any new coming user and forward message from others user to the connected user and vice versa.



Figure 2: Listen to message and send it to server



Figure 3: Handle messages

### Client

The client includes 2 parts:

* UI part:
  + Including 3 classes:
    - Frame: parent class containing some parameters: clients, port, serverIP, username, password….
    - LoginFrame: ui frame inherited from Frame class represented the Login user interface.

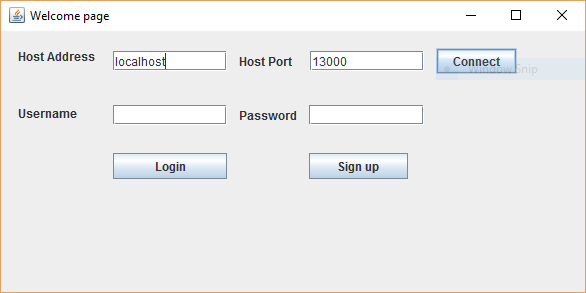


Figure 4: Login Frame

* + - ClientListFrame: ui frame inherited from Frame class represented the Chat user interface.

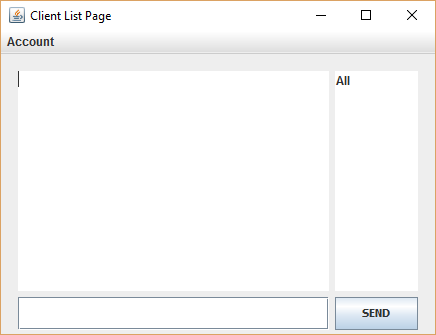


Figure 5: Client List Frame

* Socket part: including 3 classes
  + Message: data segment is wrapped and sent to server.
  + SocketClient: main class manage all the action for client. It includes the method for creating the socket connection to server by SereverIP and Port number, method for sending message and handling all the incoming message.
  + History: Class manage the message and store it in a local file. (Not finished, in progess).

The flow of running:

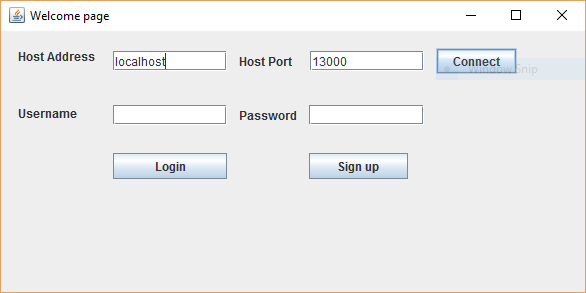
Client first connect to Server, specified by IP address and port number. Arriving messages are then displayed on message board along with their sender.

# APP running:

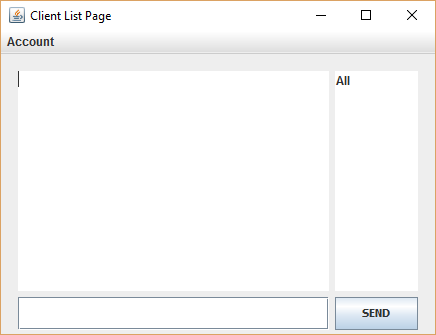
Run Server:

Run Client

* Open the ChatClient.jar



* Import the IP address and the Port number, then press Connect.
* When the connection is established, user can give the username and password. The information include username and password is send by message to Server. If there is account existed in the the system, the system will login and navigate to the Client List Frame user interface.



* User can choose the friend to chat in the list friend or choose “All” to send message to all the Client.
* User can exit the account by choosing Account/Logout.

# Code references

[1]. A java chat application: http://www.codeproject.com/Articles/524120/A-Java-Chat-Application

[2] How can I generate an MD5 hash? : http://stackoverflow.com/questions/415953/how-can-i-generate-an-md5-hash